A Kernighan-Lin Local Improvement Heuristic That Solves Some Hard Problems in Genetic Algorithms

William A. Greene

Computer Science Department, University of New Orleans
New Orleans, LA 70148, USA
bill@cs.uno.edu

Abstract. We present a Kernighan-Lin style local improvement heuristic for genetic algorithms. We analyze the run-time cost of the heuristic. We demonstrate through experiments that the heuristic provides very quick solutions to several problems which have been touted in the literature as especially hard ones for genetic algorithms, such as hierarchical deceptive problems. We suggest why the heuristic works well.

In this research, population members (chromosomes) are bit strings, all of the same length, which we denote by N. We will refer to population members as individuals.

A *local improvement heuristic* is a procedure which is applied to an individual, with the intention of modifying it into an related individual of higher fitness. Typically it is applied to a child chromosome, after its manufacture by some crossover operator but before the child is entered into the population. One form of local improvement heuristic is *hill-climbing*. One step of hill-climbing consists of flipping that bit in the individual which results in a maximal increase in fitness. To improve an individual, we might perform one step of hill-climbing, or several, or until the morphed individual has reached a local optimum and no further bit flip will increase fitness.

At a far extreme, we might argue that we do not need either a population or genetic forces at all. Any single individual differs from an optimal individual by having the wrong bit values on some subset of its bits. If we loop through all subsets of bits, each time flipping those bits in the individual, ultimately we will identify an optimal individual. But of course this approach has exponential cost, which is intolerable.

Our Kernighan-Lin local improvement heuristic falls in between hill-climbing and the extreme just described. It is inspired by the graph bisection work of Kernighan and Lin [1]. Let an individual be given. We build a tower S_1, S_2, ..., S_{MAX} of subsets of its bits. Set S_{k+1} is built from S_k by including one more bit. The extra bit is that one which is optimal with respect to change of fitness when flipped, but we allow the change to be a negative change (one no worse than the change from any other bit). We allow negative changes (hill descent), in the hope that perhaps later bit flips, in concert with previous flips, will result in yet more improvement to fitness. At most how many bits will we try flipping? Half of them ($MAX = N/2$) is a reasonable first guess. Then our heuristic actually flips those bits of whichever subset S_1, S_2, ..., S_{MAX} produces the highest fitness. A complexity analysis of our local improvement heuristic shows it to have cost $Q(N^2 g(N))$, where function g is the cost to recalculate an individual's fitness after flipping one of its bits. Function g may or may not be cheaper than the cost $F(N)$ to fully calculate the fitness of an individual that is N bits wide.

For our experiments we use a genetic algorithm that holds few surprises. It is generational, meaning that the population $P(t+1)$ at time $t+1$ is produced by creating enough

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2724, pp. 1582–1583, 2003.

children from the population $P(t)$ at time t, through crossover. It uses elitism: the best two individuals of $P(t)$ always survive into $P(t+1)$. The crossover operator is 2-point crossover. The only surprise is that fairly heavy mutation is practiced, and it is graduated, in that less fit individuals undergo more mutation. It is also stochastic, in that we make mutation attempts, but each actually results in a bit flip with only 50% chance.

We apply our local improvement heuristic first to a child after it has been created by crossover. Then we apply the heuristic again after the child has undergone mutation. Then the child is entered into the next generation $P(t+1)$. The heuristic is also applied once to the members of the initial population $P(0)$, which began as random strings.

We tried our heuristic on four problems from the literature, which have been touted as especially hard problems for genetic algorithms. The problems are: one-dimensional Ising [2], k-fold, 3-deceptive [3], hierarchical if-and-only-if (HIFF) [4], and hierarchical trap [5]. In all cases, we used a population of 40 individuals (which is smaller than in the referenced researches), and conducted 20 trials, each of which was allowed if need be to run to 500 generations. In all cases we used individuals of N bits where N was at least as wide as in the referenced researches. We experimented with various choices for MAX, the maximum number of bits which could be flipped on a local improvement step, starting at $N/2$ and decreasing. For these four problems, our approach was consistently able to find optimal individuals, in a small number of generations. Moreover, the time cost was just several seconds (using a1.7 GHz Pentium processor), when we used optimized versions of the function g. Space allows us only to report on one problem here. It will be the first hierarchical trap function of reference [5]. As in [5], individuals are 243 bits wide. In [5], costs are given in terms of fitness calculations (225,000 on average); we show our costs in terms of clocked runtimes.

Table 1. Hierarchical-Trap-#1. Size of individual = 243 bits; maximum fitness = 1215

max flippable bits	avg. final gen. #	avg. trial time (sec)	avg. best fitness
121	1.0	3.94	1215
81	1.0	2.74	1215
27	3.6	3.45	1215
9	13.55	6.29	1215
3	484.5	153.17	832.3

References

1. Kernighan, B., and Lin, S.: An Efficient Heuristic Procedure for Partitioning Graphs. Bell Systems Technical Journal, Vol. 49 (1970), pp. 291–307
2. Van Hoyweghen, C., Goldberg, D.E., and Naudts, B.: Building Block Superiority, Multimodality, and Synchronization Problems. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001), pp. 694-701. Morgan Kaufmann Publishers, San Francisco
3. Pelikan, M., Goldberg, D.E., and Sastry, K.: Bayesian Optimization Algorithm, Decision Graphs, and Occam's Razor. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001), pp. 519-526. Morgan Kaufmann Publishers, San Francisco
4. Watson, R. A., Hornby, G.S., and Pollack, J.B.: Modeling Building Block Interdependency. In: Parallel Problem Solving from Nature 1998 (PPSN V), pp. 97-106. Springer-Verlag, Berlin
5. Pelikan, M., and Goldberg, D. E.: Escaping Hierarchical Traps with Competent Genetic Algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001), pp. 511–518. Morgan Kaufmann Publishers, San Francisco